



# USAISEC

US Army Information Systems Engineering Command  
Fort Huachuca, AZ 85613-5300

AD-A268 431

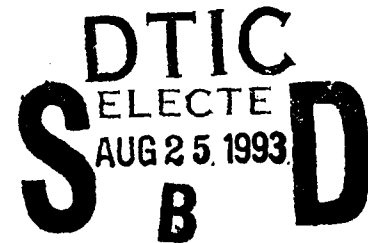
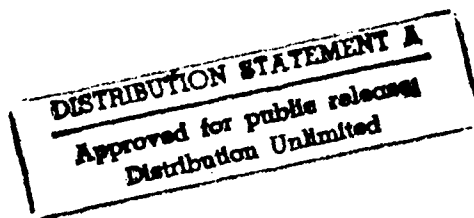


2

U.S. ARMY INSTITUTE FOR RESEARCH  
IN MANAGEMENT INFORMATION,  
COMMUNICATIONS, AND COMPUTER SCIENCES

## Multimedia Workstation Research

### Final Report



September 1990

ASQB-GC-91-003

AIRMICS

115 O'Keefe Building

Georgia Institute of Technology

Atlanta, GA 30332-0800

93 8 24 012



93-19676



25pg

## REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188  
Exp. Date: Jun 30, 1986

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS NONE		
2a. SECURITY CLASSIFICATION AUTHORITY N/A			3. DISTRIBUTION/AVAILABILITY OF REPORT  N/A		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) ASQB-GC-91-003			5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A		
6a. NAME OF PERFORMING ORGANIZATION College of Computing		6b. OFFICE SYMBOL (If applicable)		7a. NAME OF MONITORING ORGANIZATION N/A	
6c. ADDRESS (City, State, and Zip Code) College of Computing Georgia Institute of Technology Atlanta, GA 30332-0800		7b. ADDRESS (City, State, and ZIP Code)  N/A			
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AIRMICS		8b. OFFICE SYMBOL (If applicable) ASQB-GC		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c. ADDRESS (City, State, and ZIP Code) 115 O'Keefe Bldg. Georgia Institute of Technology Atlanta, GA 30332-0800		10. SOURCE OF FUNDING NUMBERS			
		PROGRAM ELEMENT NO. 62783A	PROJECT NO. DY10	TASK NO. 03-01-01	WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) (UNCLASSIFIED) Multimedia Workstation Research					
12. PERSONAL AUTHOR(S) Abe Iskac					
13a. TYPE OF REPORT		13b. TIME COVERED FROM _____ TO _____		14. DATE OF REPORT (Year, Month, Day) 1990, October 26	
				15. PAGE COUNT 20	
16. SUPPLEMENTARY NOTATION					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUBGROUP	multimedia workstation, ISDN, B-channel, D-channel, video conferencing, LAPD protocol		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)  Current ISDN implementation strategy requires each end-user's equipment to provide suitable types of terminating devices in order to access the ISDN networks. This project attempts to investigate the feasibility of integrating all ISDN TES in a multimedia environment.					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL Winfred Y. Fong			22b. TELEPHONE (Include Area Code) (404) 894-3136		22c. OFFICE SYMBOL ASQB-GC

This work is done under contract DAAL03-86-D-0001 for the United States Army Institute for Research in Management Information, Communications, and Computer Sciences (AIRMICS), the RDTE organization of the United States Army Information Systems Engineering Command (USAISEC). The principal investigator was from the College of Computing, Abe Iskac of the Georgia Institute of Technology.

This report is not to be construed as an official Army position, unless so designated by other authorized documents. The material included herein is approved for public release, distribution unlimited, and is not protected by copyright. Your comments on all aspects of the document are solicited.

**THIS REPORT HAS BEEN REVIEWED AND IS APPROVED**

<b>Accession For</b>	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

s/ \_\_\_\_\_  
John W. Gowens  
Division Chief  
CNSD

s/ \_\_\_\_\_  
John R. Mitchell  
Director  
AIRMICS

**DTIC QUALITY INSPECTED 3**

# Multimedia Workstation Research

by

Abe Iskac

College of Computing  
Georgia Institute of Technology  
Atlanta, Georgia

for

AIRMICS  
Army Institute for Research in Management  
Information, Communications, and Computer Sciences  
Atlanta, Georgia

October 26, 1990

Contract No. DAAL03-86-D-0001

Delivery Order 2166

Scientific Services Program

The views, opinions, and/or findings contained in this report are those of the author and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

## TABLE OF CONTENTS

	Page
List of Figures	1
1 Introduction	2
2 Technical Discussion	3
2.1 Setting Up Hardware and Software Development Environment for Conducting Research in Multimedia Workstation	5
2.1.1 Hardware Setup	5
2.1.1.1 Multimedia Development System Rack	5
2.1.1.2 Power Supply Unit for the rack	5
2.1.1.3 Prototype Cards	6
2.1.1.4 Backplane	6
2.1.1.5 Test Equipment	6
2.1.2 Software Setup	6
2.1.2.1 OrCad Schematic Capture	6
2.1.2.2 68010 C-Compiler and Assembler	6
2.1.2.3 Eprom Programmer	6
2.1.2.4 Serial Loader Software	7
2.2 Design the Architecture of a Multimedia Workstation	7
2.3 Design and Implement the Individual Processing Modules, Firmware for each Processing Module, and System Software to tie all the Processing Modules together	8
2.3.1 Design, Implement and Test the IOFEP Module	9
2.3.2 Design, Implement and Test the VOP Module	12
2.3.3 Design, Implement and Test the VCP Module	13
2.3.4 Design, Implement and Test the Shared Memory Mechanism on the VOP Module	14
2.3.5 Design, Implement and Test the Signaling Mechanism between the IOFEP and VOP Modules	16
2.4 Design the User-Interface Software	19
3 Conclusion	20
4 Recommendations	20

## **LIST OF FIGURES**

Figure 1. Rack Assembly for the Multimedia Workstation System

Figure 2. Multimedia Workstation System in a self-contained unit

Figure 3. Block Diagram of IOFEP Module

Figure 4. Block Diagram of VOP Module

Figure 5. Block Diagram of VCP Module

## 1 INTRODUCTION

With the introduction of Integrated Services Digital Network - ISDN, users are given much higher bandwidth to communicate than provided by the conventional analog line. Along with the increase in throughput, the information is to be carried in digital form.

These two aspects of ISDN create on the one side the opportunity to supply more services on the existing communication network, while on the other side generate a collection of terminating equipment to support them.

While it is relatively simple to operate even the most sophisticated analog telephone with its DTMF (Dual Tone Multiple Frequency) features, it is not so straightforward to make use of an ISDN telephone, that shares the D-channel with an ISDN PC card, each sharing different TEI numbers (Terminal Equipment Identifier).

The above example gets more complicated if more terminating equipment are to be added in a user node (up to 8), such as the case if the need to transfer video information arises.

The trend in the development of ISDN terminating equipment has been application specific, which means that in general there is one interface card to do data transfer and there is one analog unit to do voice transfer, with both units physically separated. This architecture creates not only difficulty in keeping up with the operation of each unit in the system, but also with the physical operation of each distributed unit in the work environment.

The following research work takes as an input the above distributed terminating equipment in an ISDN work environment and proposes a solution in terms of integrating them into a Multimedia Workstation for ISDN.

It further designs and implements a prototype of multiple processing engines (modules) for the workstation unit, which are to be capable to handle all the tasks needed in an ISDN Multimedia Environment.

This phase of the project is concluded by the test and evaluation of the processing engines, both independently and as a unit, that satisfies the requirements of an ISDN Multimedia Workstation.

## 2 TECHNICAL DISCUSSION

The research work is intended to study the feasibility of integrating ISDN terminating equipment for different applications into one self-contained unit. A big stress is put on the integration of video information transfer into the system, allowing video conferencing to take place over ISDN. While the first design and implementation phase will touch only uncompressed video data, thus transmission of still video, the mechanism should be provided to transmit compressed video information in the future.

The goal of the research work is to propose a single-unit solution to do all the tasks required in an ISDN environment. The following tasks are representable for a workstation's day to day operation in such an environment :

- Voice Call on one of the ISDN B-Channel.
- Data transfer on one of the ISDN B-Channel.
- Video/Image transfer on one or both of the ISDN B-Channel.
- Local PC/Workstation operation, such as compiling a program.
- Retrieve graphics document to be sent out through one of the ISDN B-Channel.
- Open a window on a PC/Workstation monitor to start video conferencing session.

To achieve the processing power required to handle all the above tasks, an architecture consisting of three tightly-coupled processing modules is designed and prototyped. The selection of three independent processing modules is based on the tasks discussed above.

The first group of tasks consisting of ASSIGNING and ACCESSING the channel in the ISDN interface, PROCESSING the LAPD protocol on the D-Channel, and COMMUNICATING with the PC/Workstation, requires the processing power of one microprocessor. A single Motorola MC68010 running at a clock speed of 12 MHz is enough to handle all these tasks in real-time. Since this microprocessor has a serial port to communicate with the host, it is the main processing module of the system, and it is designed as the master of the whole system. This first processing module is called IOFEP - Input Output Front End Processor.

The second group of tasks deals mainly with the management of hardware windows on a standard PC/Workstation monitor for the purpose of displaying real-time video information, such as data taken from a video camera and/or received from the ISDN B-channel(s). Since this information needs to be displayed in real-time on the same screen of the PC/Workstation, a high processing power is required. This dedicated processing power can be delivered by a second MC68010 with the main function of managing and overlaying video data on top of the standard video signal. This second processing module is called VOP - Video Overlay Processor.

The third group of tasks concentrates on capturing the image from a video camera or standard NTSC video source. A third MC68010 microprocessor is allocated to do this

task. While most of the video capture mechanism is done in logic, the microprocesor is useful in further processing of the video data, such as compression and decompression in the future. This third processing module is called VCP - Video Camera Processor.

While each processing module represents a stand-alone single-board microcomputer complete with on board memory and firmware, the whole system has to work together as one unit. The research work defines what each processing module is, how it has to look like independently, and how it has to interact with one another to provide hardware and software transparency.

To support the research and development effort, a laboratory for the AIRMICS specifically for this project needs to be set up, both in terms of hardware and software development.

The project is broken down into the following stages :

- Setting up the hardware and software development environment for conducting research in Multimedia Workstation.
- Design the architecture of a Multimedia Workstation.
- Design and Implement the individual processing modules, firmware for each processing module, and system software to tie all the processing modules together.
- Design the User-Interface-Software

## **2.1 SETTING UP THE HARDWARE AND SOFTWARE DEVELOPMENT ENVIRONMENT FOR CONDUCTING RESEARCH IN MULTIMEDIA WORKSTATION**

The following tasks were completed as a preliminary requirement to conduct hardware and software development for the main research project.

### **2.1.1 HARDWARE SETUP**

The following preliminary equipment were needed and setup accordingly (s. figure 1) :

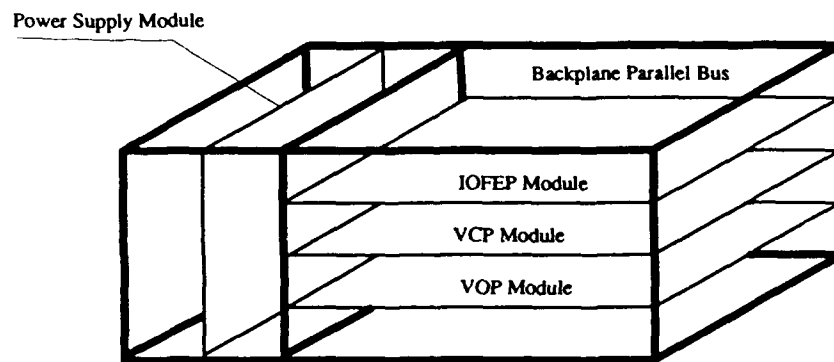


Figure 1. Rack Assembly for the Multimedia Workstation System

#### **2.1.1.1 Multimedia Development System Rack**

This is a 19" wide standard system rack providing three horizontal slide-in card slots and one vertical slide-in card for the rack power supply unit.

#### **2.1.1.2 Power Supply Unit for the rack**

A 10A minimum, 5 V single output, regulated switching power supply unit mounted on the vertical slide-in card of the rack. The 5 V output is distributed to the backplane of the system rack.

### **2.1.1.3 Prototype Cards**

Three blank perforated prototype cards for the horizontal slide-in slots of the rack. Each prototype card carries two 96-pin male edge connectors to communicate with the other cards through the backplane.

### **2.1.1.4 Backplane**

The backplane contains the matching 96-pin female edge connectors for each horizontal slots. It distributes the 5 V power from the output of the power supply unit to each horizontal prototype card. It also provides external bus connections for each prototype card to communicate with the others.

### **2.1.1.5 Test Equipment**

At this stage of the development, a logic probe and a digital voltmeter are available for use.

## **2.1.2 SOFTWARE SETUP**

Software setup is conducted through the installation and testing of the following items :

### **2.1.2.1 OrCad Schematic Capture**

All circuits implemented as part of the prototype system are drawn in OrCad Schematic Capture running on an IBM/PC-XT.

### **2.1.2.2 68010 C-Compiler and Assembler**

Aztec 68010 C-Compiler running on an IBM/PC-XT is to be used to generate the codes for the three system boards of the Multimedia Workstation System. When a timing-related program execution or a fast response time is required, the codes will be written in assembly language, and the Axtec 68010 Assembler will be used. A loader program is required to transmit the object codes to the EPROM programmer.

### **2.1.2.3 Eprom Programmer**

A stand-alone EPROM programmer with a serial asynchronous port will be attached to an IBM/PC-XT, where the software development system resides. Codes generated by the compiler or assembler will be dumped serially to the EPROM programmer, and will be

burned into the EPROM chip.

#### **2.1.2.4 Serial Loader Software**

A program to read in the object code from the compiler/assembler output file and to send out the byte one character at a time to the asynchronous serial port, which in turn is connected to the EPROM programmer. In the advanced stage, where the system software is to be loaded directly to the system RAMs, this Serial Loader Software will communicate directly with the monitor program of the main system board.

## **2.2 DESIGN THE ARCHITECTURE OF A MULTIMEDIA WORKSTATION**

A Multimedia Workstation for ISDN has to have the following major functionalities :

- capability to input and output voice in real-time.
- capability to display video image in the hardware window of its standard PC/Workstation monitor.
- capability to communicate with the ISDN network.

Based on the above requirements, a powerful processing system is needed. It is not enough to share one microprocessor to do all the tasks mentioned above. Thus, a system architecture consisting of three independently running processing modules is necessary to achieve the goal of a real-time Multimedia Workstation System.

While the demand on the real-time voice input and output can be achieved rather easily, the other requirements are not very trivial to accomplish and will present significant demands on the processing power of the whole system. To achieve a real-time display of video information together with some other standard text/graphics information on the same screen, an independent processing module called VOP - Video Overlay Porcessor is needed.

Function of the VOP module is to overlay the standard video signal originating in the video controller board with a real-time video information either from the ISDN line or the video camera, before it is sent to the screen. This display overlay function is done independently of the other functions in the system.

While most of the overlay functions can/will be done in hardware, a micro- processor is needed to pass information to the hardware such as beginning and end of the video window, as well as communication with the main processor, the IOFEP module.

Another high processing power is required to digitize the real-time video input from the video camera and to process it. Processing of this video signal involves loading it to the VOP directly and/or sending it to the IOFEP for the purpose of storing this video

information in the host. All these functions have to be done by another independent microprocessor, the VCP - Video Camera Processor module.

The ISDN interface layer 1, as well as the telephone handset interface will be done in the hardware of the IOFEP, and requires very small processing power of the microprocessor. Most of the processing power of the IOFEP microprocessor is needed to do the following tasks :

- collect and send data from and to the VOP or VCP module.
- control LAPD processing of the ISDN D-Channel.
- control high-speed X.25 communication with the host computer.

From the above requirements, a three tightly-coupled processor system as shown in figure 2 is the most suitable approach to the architecture of a Multimedia Workstation System.

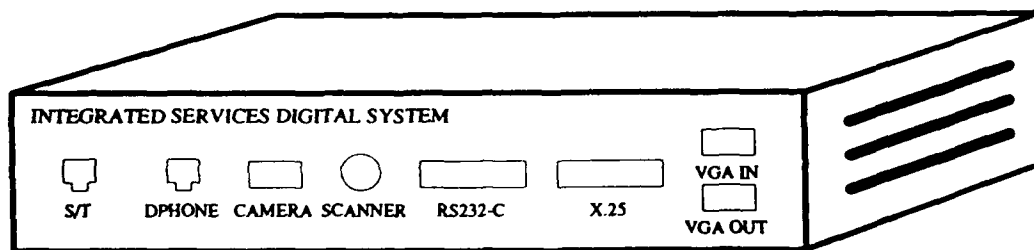


Figure 2. Multimedia Workstation System in a self-contained unit

### **2.3 DESIGN AND IMPLEMENT THE INDIVIDUAL PROCESSING MODULES, FIRMWARE FOR EACH PROCESSING MODULE, AND SYSTEM SOFTWARE TO TIE ALL THE PROCESSING MODULES TOGETHER**

For the purpose of debugging the processing modules independently of each other, and since each module is a fully functional microcomputer, the design and implementation of the system is done in the following steps :

1. Design, Implement and Test the IOFEP module.
2. Design, Implement and Test the VOP module.
3. Design, Implement and Test the VCP module.
4. Design, Implement and Test the Shared Memory Mechanism on the VOP.  
Design, Implement and Test the Shared Memory Mechanism on the VCP.
5. Design, Implement and Test the Signaling Mechanism between the IOFEP and VOP.

## Design, Implement and Test the Signaling Mechanism between the IOFEP and VCP.

### 2.3.1 DESIGN , IMPLEMENT AND TEST THE IOFEP MODULE

After the design of the microcomputer for the IOFEP Module is completed, it is captured in OrCad, and wiring of the microcomputer section of the IOFEP module is started. See the block diagram in figure 3.

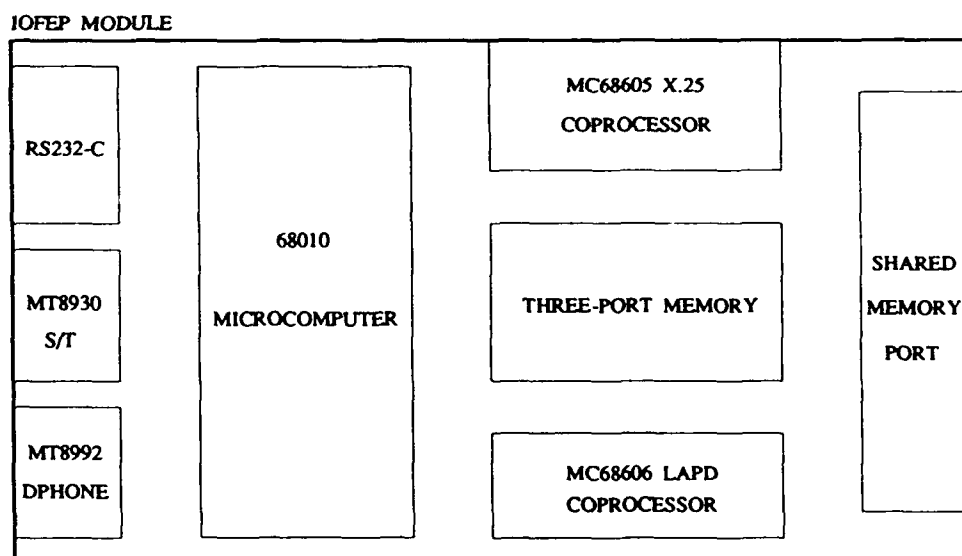


Figure 3. Block Diagram of IOFEP Module

Functional testing is performed as each block is wired. The following steps are used as guidelines for the microcomputer implementation. First, the microprocessor control signals are wired and its functionalities are tested. Then, the rest of the hardware are implemented and tested in the following order :

- Memory Control Logic
- Memory Devices (64 Kbytes EPROMs and 0.5 Mbytes Dynamic RAMs)
- Clock Generator (24 MHz main clock and divider)
- Reset Logic
- Bus Error Logic

At this stage of implementation, a test program needs to be burned into the EPROM, which requires an external EPROM programmer to be attached and configured, to run

under the IBM PC control. Preparations need to be made to burn this monitor program into the EPROMs for the IOFEP Module. This object code will be the firmware that handles the user interface to the whole Multimedia System.

A PAL (Programmable Array Logic) device is also needed at this stage and programmed accordingly to do the function of a memory decoder. This is to reduce the number of discrete components used in the hardware to achieve the same function.

After the microcomputer section of the IOFEP module is finished and tested, the following tests were performed step-by-step to conclude that the microcomputer hardware functions properly :

### Test1

A very small program was burned into the EPROMs to examine that the microprocessor can fetch and execute the object codes without error. If the microprocessor, the bus wiring and all signal conditions are correct, the program will cause the CPU to stay in an infinite loop, which can be checked very easily with a logic probe by the presence of pulses in all the circuits. In other case, the CPU will halt and drive the HALT signal pin low.

### Test2

The same test above was added with a CALL to a Dynamic Memory Test Routine. This test examines the integrity of the RAM. An error in the test causes an ILLEGAL instruction to be executed, which in turn makes the processor halt.

### Test3

A serial asynchronous console port is built as part of the IOFEP module. This console port was examined in this test. The main TEST1 above was added with two procedure CALLS:

- initialize the asynchronous device after reset.
- echo test routines, allowing input/output from/to the host computer (IBM PC/XT) through the RS232 interface.

TEST1 is used as the initial loop causing the microprocessor to stay in a controllable state, and to which more procedure calls are added.

After successful completion of the tests above, it was concluded that the microcomputer section of the IOFEP module works perfectly and ready to be used.

The monitor program, written and tested previously in the RAM of an existing 68000 based microcomputer, is now burned into the EPROMs of the IOFEP microcomputer. Aside from the fact that the code has to reside in a different address space and thus requiring it to be linked at a different starting address, everything else is left unchanged.

This version of the monitor program is referred to as "AIRMICS-ISDS Monitor 1.0".

An LED register is added and functionally tested. Furthermore, an 8-bit DIPSWITCH is implemented as part of the IOFEP peripheral devices. Both registers share the same physical address, with the LED register acts as a write-only and the DIPSWITCH register as a read-only register.

An LED register is a one byte addressable output latch, accessible from the microprocessor through the execution of a 'MOVE.B #data,LEDREG' instruction. A one byte LED register is usually a requirement for a single board microcomputer to display program execution activities (i.e. certain program condition may cause a specific bit pattern to be output to the LED under program control).

A DIPSWITCH register is a one byte addressable input buffer, accessible from the microprocessor through the execution of a 'MOVE.B DIPSWITCH,d0' instruction. A one byte DIPSWITCH register, selectable from the user side by setting its bit position to '0' or '1', is also a requirement for a single board computer as a way of setting the operation mode of the board. This modes can be decoded as bit patterns, that will be checked by the microprocessor on power-up or regularly as part of the system task check. This way, the system can be forced to enter a certain mode of operation such as debugging mode or operation mode. This input register will be a big help during the software development phase of the system.

After the monitor program is run successfully off the EPROMs, an S-Record Loader is added to the monitor program - written in C.

An S-Record Loader is part of the monitor (firmware) residing in the EPROMs of the microcomputer board to allow loading of object codes from a host development computer (eg. IBM PC/XT-AT) through a standard communication line. In the IOFEP module, this communication line will be an RS232 port. In the other processing modules (VOP and VCP), it will be a parallel port accessible through the parallel bus on the backplane.

An S-Record Loader accepts only an object code file conforming to the Motorola S-Record format.

Using the built-in S-Record Loader (in the firmware), an additional feature is added to the monitor allowing detection and processing of an external bus error signal. The S-Record Loader speeds up the cycle of testing this feature by allowing the codes generated by the compiler to be loaded into the RAM and then executed from there, without the need to erase and burn new EPROMs.

Detection and processing of an external bus error signal is needed to terminate an access (read or write access) to a non-existing memory. Every microprocessor access to a memory location is timed by a hardware logic called "Watch-Dog Timer". If the memory being accessed does not respond within a certain period of time (system clock dependent, usually in the microseconds range), an active pulse is generated by the Watch-Dog Timer to the Bus Error Input of the microprocessor.

### 2.3.2 DESIGN, IMPLEMENT AND TEST THE VOP MODULE

After the first microcomputer module is implemented and functionally tested, wiring of the second module, the VOP (Video Overlay Processing) Module is started. See figure 4.

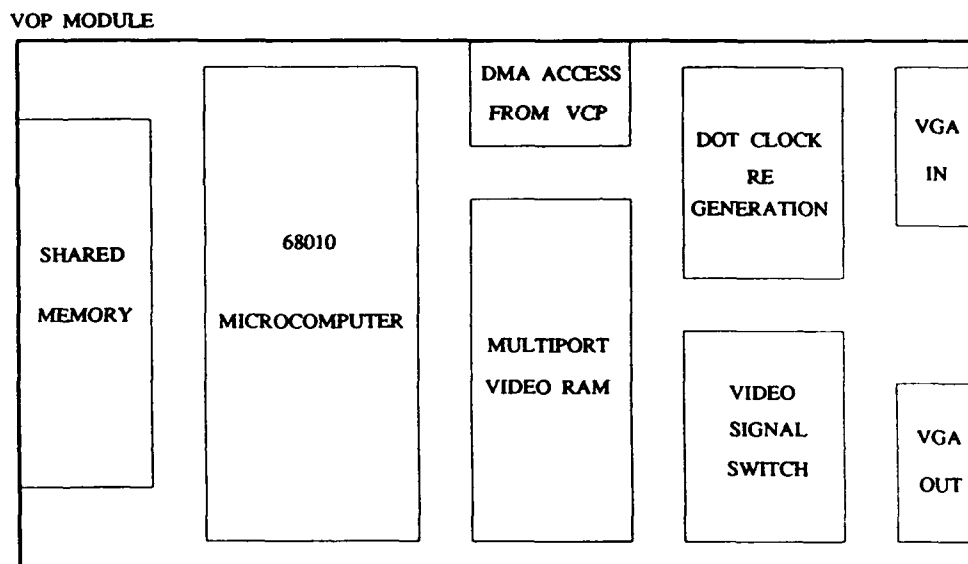


Figure 4. Block Diagram of VOP Module

The implementation and test phases follow exactly the ones of the first module. It has everything that the IOFEP has, with the exception of an asynchronous RS232 port. A program to run memory test is burned into the EPROMs of this VOP module and run continuously with no error. The same test procedures as described earlier for testing the hardware functionality of the IOFEP module were performed here step-by-step. (Any hardware or software errors would cause the VOP microprocessor to halt).

Since the VOP module shares half of its on-board memory with the IOFEP module, design and implementation of the backplane interface needs to be started at this point for the purpose of accessing the VOP memory from the IOFEP module.

Initially, a parallel port was chosen to perform this task, with the capability to emulate the serial port of the first module. This design is enhanced by incorporating it through the shared memory on the VOP module. This reduces the number of hardware components

needed on each board to achieve the same result, while at the same time provides a better command passing mechanism from one board to another.

An 8-bit LED register and DIPSWITCH register are added to the VOP (Video Overlay Processor) module and functionally tested. The VOP's LED register at this point of implementation is necessary to confirm the functionality of the shared memory residing on the VOP module. The test was conducted as follows :

The VOP microcomputer is running a program independently off its EPROM to output incrementing values to its on-board LEDs. To run the program, it uses an integer value (16bit) as a constant for a delay function, and this value is specifically stored in the area of the shared memory. Without any access from the IOFEP module to the shared memory, the above LED test program runs continuously, confirmed by the change of LED patterns in an incrementing fashion.

If the design and implementation of the shared memory access is to be confirmed, the LED test program should continue running, even with a continuous IOFEP access (read/write) to the shared memory. Furthermore, the IOFEP should be able to change the integer value used as a delay constant by the LED test program while it is running, which can be seen by the frequency of change of the LED patterns.

At this point it is necessary to add a "RESET" command to the monitor on the IOFEP module to allow software reset to the on board devices as well as the other two microcomputer boards without resetting the local microprocessor.

### **2.3.3 DESIGN, IMPLEMENT AND TEST THE VCP MODULE**

The third microcomputer called VCP - Video Camera Processor - is a copy of the second microcomputer module and is completed as well as functionally tested in a short period of time. See figure 5.

The same test procedures as previously used for the IOFEP and VOP modules are performed here step-by-step to check the functionality of the VCP module. The dynamic memory chips (TMS44C256-10) were borrowed from the second board to do the above testings. To further test the shared memory of the VCP module all the backplane wiring has to be done, allowing all the address, data and signal buses to reach the third board.

VCP MODULE

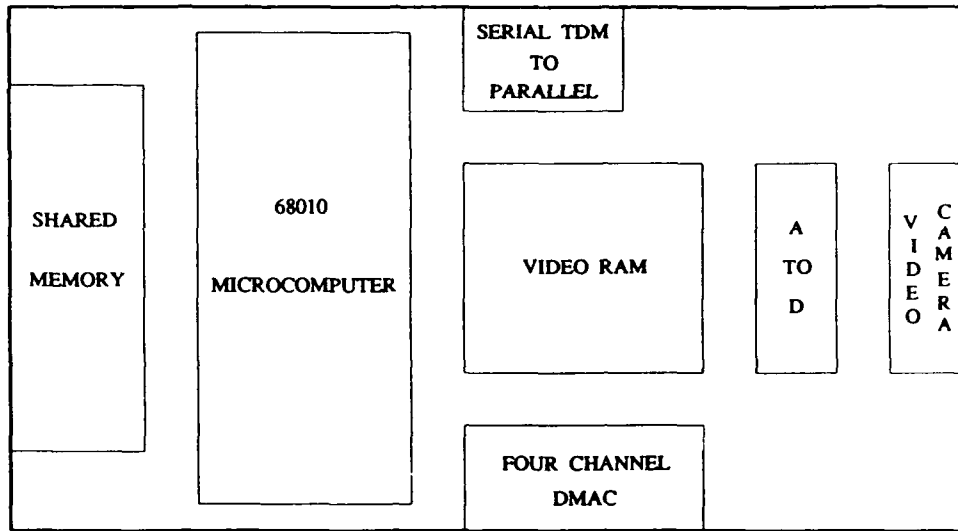


Figure 5. Block Diagram of VCP Module

#### 2.3.4 DESIGN, IMPLEMENT AND TEST THE SHARED MEMORY MECHANISM ON THE VOP MODULE

The shared memory mechanism is designed and implemented. After several design modifications in conjunction with the test procedure and expectations, the signal generations to access the shared memory on the VOP module are confirmed.

In the implementation of the shared memory on the VOP module, the local VOP memory is shared. A certain amount of protection is implemented in hardware, in such a way that the local memory of the VOP module is split in two halves. Only the upper half can be read and/or written by the IOFEP module. This keeps local sensitive codes or data from being changed by other module, providing they are stored in the lower half of the memory. Each half is 256 kbytes.

Due to the complexity of the tightly-coupled microcomputer system, a hardware reset from the main IOFEP module is added to reset all other modules in the system, in case one or more modules get hung up.

Problem with the shared memory access causing an occasional hanging of the IOFEP, encountered at the first stages of implementation, is now corrected. The problem was caused by disabling the backplane interface chips too early, while the IOFEP microprocessor was still in the process of reading or writing from/to it.

The shared memory is located on the VOP board, which is the second board in the system. To reduce the amount of components and to facilitate better communication mechanism between the IOFEP and VOP, half of the local RAMs of the VOP module is shared, providing 256 kbytes of shared memory locations. The RAM layout of the IOFEP and VOP modules is as follows :

module	address range	memory access
IOFEP	00000000	local
	0007FFFF	
VOP	00000000	local
	0003FFFF	
	00040000	local and shared with IOFEP
VCP	0007FFFF	local and shared with IOFEP
	00000000	
	0003FFFF	
	00040000	
VOP		local
	0007FFFF	
	00000000	
	0003FFFF	
VCP	00040000	local and shared with IOFEP
	0007FFFF	
	00000000	

\* the IOFEP addresses this shared memory portion by adding 0x100000 to the physical address of the VOP memory, making it in the range of 00140000 - 0017ffff

\*\* the IOFEP addresses this shared memory portion by adding 0x200000 to the physical address of the VCP memory, making it in the range of 00240000 - 0027ffff

A very extensive shared memory test over an extended period of time ( overnight) is performed. Data integrity, as well as control signals are tested and confirmed for their functionalities.

### 2.3.5 DESIGN, IMPLEMENT AND TEST THE SIGNALING MECHANISM BETWEEN THE IOFEP AND VOP MODULES

The next logical step in the implementation of the system is to run the monitor program to access the VOP and the VCP modules, all through a command passing protocol through the shared memory on each of the above modules.

To do this, hardware interrupt mechanism is designed and implemented. The first implementation and test are performed on the IOFEP module. This interrupt hardware allows up to seven prioritized devices to get the cpu attention by generating an interrupt signal to it. The first device used for testing is the RS232 console port. The port is programmed in such a way that each new character received from the keyboard causes an interrupt to be posted. The interrupt mechanism is tested successfully.

Interrupt mechanism is needed for the implementation of command passing protocol between the IOFEP and VOP. The idea is as follows :

If the IOFEP needs to send a command to the VOP, the command will be packetized and moved to the predefined shared memory location of the VOP. This command may be one block of 64 bytes or more, depending on the nature of the command. Once the full block is loaded, a signal will be generated by the IOFEP to interrupt the VOP. This way, the VOP can be signalled of the availability of a command block for processing. The same mechanism will be used in the other direction (VOP to IOFEP).

Another source of hardware interrupt, an Event Clock Interrupt, is added to the IOFEP module, and assigned a priority level 6. As it stands now, the interrupt priority scheme looks as follows :

interrupt priority level	interrupting device
--------------------------	---------------------

7	ABORT Switch (not wired)
6	60 Hz Event Clock
5	VOP Packet Available (not wired)
4	VCP Packet Available (not wired)
3	not assigned
2	console port
1	not assigned

A level 7 interrupt is a non-maskable interrupt originating in a system's fault-sensitive device that needs an immediate attention from the microcomputer. In our system, this signal will come from a push-button, which when pressed will cause the microcomputer to abort any program execution being performed and bring the control back to the monitor. This is a necessary feature in debugging the software, since it will allow saving the context of the program at the time the ABORT signal was generated.

A level 6 interrupt comes from a programmable clock generator, which is programmed to supply a continuous 60 Hz clock. This Event Clock is used as the basic mechanism of Context-Switching and Multitasking.

Level 5 and 4 are dedicated for signalling control between the IOFEP and VOP's/VCP's shared memory.

Level 3 and 1 are allocated for future devices, such as the ISDN interface controller.

Level 2 is assigned for the RS232 console port. This is only available on the IOFEP module.

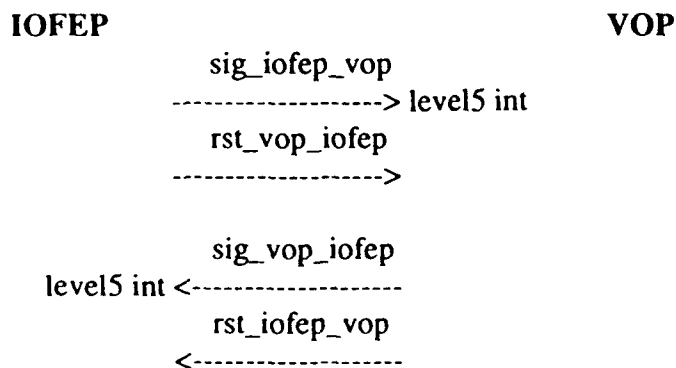
The Event Clock interrupt is tested in such a way, that every 60 interrupts cause LED #0 to blink. An Event\_Clock\_Handler is added to take care of counting the events and toggling the LED bit.

Test is performed on the IOFEP to enable both console port and event clock interrupts to confirm the integrity and functionality of the IOFEP module. All activities are shown on the LEDs as well as on the monitor console.

The level 5 interrupt on the IOFEP and VOP modules to do packet level signalling between the above two modules, are now implemented and tested. The test was performed as follows :

A signalling hardware is added to the IOFEP and VOP modules. This signalling hardware is accessible from each microcomputer as output port, and it has the function of generating a pulse to the other module (through software control).

Through output of different data bytes to these ports, two independent pulses can be generated. The first pulse is to generate an interrupt signal and the second pulse is to clear the cause of interrupt from the other module (s. diagram below !).



The signal `rst_vop_iofep` is generated to clear the level5 int from VOP caused by `sig_vop_iofep`.

To test the above signalling mechanism, a `level5_handler` routine is added to the monitor on each module. This handler will be invoked by the microcomputer whenever a level5 int is acknowledged.

Inside the `level5_handler`, an LED is blinked and the corresponding reset pulse is generated. Before returning from this handler, a signal to the other module is generated. This way, the signalling mechanism will run continuously by itself and its functionality and integrity can be made visible by the blinking of LED on each module. This test is performed successfully.

With the test just described, all interrupt sources on IOFEP module are now enabled:

- Level6 ... 60 Hz Event Clock
- Level5 ... Packet Signalling Interrupt
- Level2 ... RS232 Console Input

The test is to confirm the overall design and implementation of the interrupt logic. The interrupt handlers of each interrupt source blink a different LED, in addition to performing the real service routines. The test runs successfully.

At this point all the hardware functions of the prototype unit have been successfully tested. The next step is to utilize this hardware, such as the shared memory and signalling hardware, to provide transparent access from the user to each processing module. Packet and Device Drivers are written for this purpose and functionally tested. The test procedures are as follows :

Characters received from the main RS232 port are packetized and received by the RS232 driver on the IOFEP module through interrupt level 2. This packet is sent to the specified shared memory area on the VOP module by a packet driver on the IOFEP, accompanied by an interrupt signalling at level 5.

The Level 5 packet handler depacketizes this packet and further passes it on to the command interpreter on the VOP module.

The VOP has the same monitor as the IOFEP, which interprets the incoming command string from the shared memory and sends the result(s) back to the shared memory with the help of the packet handlers. The end destination of the packet(s) is the same RS232 port on the IOFEP module.

The test above is conducted successfully and is used as the basis for the implementation of a transparent communication with all the processing modules in the prototype system. With a slight software modification, the transparent access to the VOP is achieved.

With the console communicating transparently with the VOP module (through the IOFEP module), all monitor commands are executed successfully.

At this point, all design, implementation and test phases planned at the beginning of the project have been successfully performed. An additional work is done in the user interface area, which is reported in the User-Interface-Software section below.

## **2.4 DESIGN THE USER-INTERFACE-SOFTWARE**

The software development environment for the Multimedia Workstation Research is accomplished, but a better interface windowing mechanism would be preferable. For this purpose, a communication software with multiple windowing mechanism specifically tailored for this project is designed and implemented.

Initial coding in C is started and tested. This is a software package running currently on an IBM PC that has the function of a Window Manager. This interface software allows the user to open several windows on a PC monitor and to assign each window as the I/O device to each processing module in the system. By pressing specific function keys, the user can go from one processing module to another.

This initial windowing user interface software is a requirement to increase the productivity in the software development of the prototype system.

### 3 CONCLUSION

The Multimedia Prototype System as a unit consisting of three independent processing modules is designed and functionally tested. All the small hardware and software pieces, such as the signalling logics, shared memory drivers, interrupt handlers, etc. fit and work together, making the multiple processor system look like a single powerful and totally programmable microcomputer system.

The Multimedia Workstation Architecture is designed based on three major tasks requiring special processing power. They are as follows :

- capability to input and output voice in real-time.
- capability to display video image in the hardware window of its standard PC/Workstation monitor.
- capability to communicate with the ISDN network.

Based on the above requirements, a powerful processing system is needed. It is not enough to share one microprocessor to do all the tasks mentioned above. Thus, a system architecture consisting of three independently running processing modules is necessary to achieve the goal of a real-time Multimedia Workstation System.

We have prototyped and functionally tested the system, both individually and as a system and have shown that the architecture satisfies the requirement to provide enough processing power for a multimedia workstation.

A software debugging tool is developed, both for an individual board and for the system as a unit, to facilitate further system and application software development.

### 4 RECOMMENDATIONS

Further development of the system should be performed in the following order :

1. Develop Video Overlay Capability on the VOP module.
2. Develop ISDN S-Interface on the IOFEP module.
3. Develop LAPD (soft-and hardware) for the ISDN S-Interface.
4. Develop Video Input Interface on the VCP module.
5. Develop high-speed X.25 Interface between the PC and the IOFEP.
6. Develop Application Software for the system.